

Quotients in monadic programming: Projective algebras are equivalent to coalgebras

Dusko Pavlovic*

Peter-Michael Seidel

University of Hawaii, Honolulu HI, USA

Abstract

In monadic programming, datatypes are presented as free algebras, generated by data values, and by the algebraic operations and equations capturing some computational effects. These algebras are free in the sense that they satisfy just the equations imposed by their algebraic theory, and remain free of any additional equations. The consequence is that they do not admit quotient types. This is, of course, often inconvenient. Whenever a computation involves data with multiple representatives, and they need to be identified according to some equations that are not satisfied by all data, the monadic programmer has to leave the universe of free algebras, and resort to explicit destructors. We characterize the situation when these destructors are preserved under all operations, and the resulting quotients of free algebras are also their subalgebras. Such quotients are called *projective*. Although popular in universal algebra, projective algebras did not attract much attention in the monadic setting, where they turn out to have a surprising avatar: for any given monad, a suitable category of projective algebras is equivalent with the category of coalgebras for the comonad induced by any monad resolution. For a monadic programmer, this equivalence provides a convenient way to implement polymorphic quotients as coalgebras. The dual correspondence of injective coalgebras and all algebras leads to a different family of quotient types, which seems to have a different family of applications. Both equivalences also entail several general corollaries concerning monadicity and comonadicity.

1 Introduction

1.1 The story

Background: Monadic programming. Monads are one of functional programmers' favorite tools, and possibly one of the most popular categorical concepts [14, 13]. As a type constructor, a monad gives rise to datatypes that capture not only the data values, but also some computational effects of interest [35, 42]. While this is achieved using

*Supported by AFOSR and NSF.

a very simple and convenient set of tools, the history of the underlying ideas is convoluted, and the conceptual and technical background of monadic programming covers enough territory of algebra and of category theory to conceal many mathematical mysteries.

The conceptual origin of monadic programming was probably the idea that data structures can be captured as algebraic theories, which goes back to the early days of semantics of computation [20, 21]. The technical origin of monadic programming was then the idea that algebraic theories can be captured as monads, which goes back even further, to the early days of category theory [33, 34, Ch. I]. The upshot of the view of data-structures-as-algebraic-theories is that computational datatypes, as domains of inductive and recursive definitions, can be viewed as *initial*, or *free* algebras, implementing induction as a universal property. The upshot of the view of algebraic-theories-as-monads in this context is the fact that monads encapsulate and hide behind their standard structure¹ the diverse and often bewildering sets of algebraic operations, and make them available only through uniformly structured monadic combinators.

The main feature of computational monads is thus their succinct and elegant rendering of inductive datatypes as *free* algebras. But this main feature is perhaps also their main limitation: the quotients of free algebras are not free algebras.

Problem: Quotient types. Whenever a data value can be given by different representatives, its datatype is a quotient. E.g., each rational number can be represented by infinitely many fractions ($\frac{3}{7} = \frac{39}{91} = \frac{273}{637} = \dots$), so the datatype of rationals is a quotient of the datatype of ordered pairs of integers. Sets are a quotient of bags, bags are a quotient of lists, and so on. Identifying the equivalent representatives can be a hard and important computational task, tackled in type theory from the outset, going back to Martin-Löf, and still further back to Leibniz. Different applications often justify different implementations [1, 6], which vary from simply carrying explicit equivalence relations with *setoids* [11, 31], through carrying *coherent* equivalences with *groupoids* [25, 24, 3], all the way to the rich structure of *homotopy types* [8, 40], where the problem of quotients in type theory and the problem of invariants in geometry seem to be solving each other.

The basic idea of monadic programming, to present datatypes as *free* algebras, precludes direct implementations of quotient types, since a quotient of a free algebra is in general not free. This is often viewed as a feature, since polymorphism requires that all data satisfy exactly the same equations, which for algebras means that they should satisfy just the equations imposed by their algebraic theory, and no additional equations. When necessary, the additional equations can be imposed by explicit destructors, but the polymorphic constructions generally do not carry over to such quotients, unless the destructors preserve them. Under which conditions does that happen?

Solution: Projective algebras. In the present paper, we study a special family of quotients of free algebras: those that also happen to be their subalgebras. This means that they can be implemented not only by imposing additional equations, but also by

¹Godement introduced monads under the name *standard construction*, standardizing the sheaf cohomology construction [19].

adjoining suitable operations, called *projectors*, as described below. An algebra which is both a quotient and a subalgebra of a free algebra, i.e. its retract, is said to be *projective* [22, §82]². Since projectors induce precisely the quotients that are preserved by all functors [36], this approach seems necessary and sufficient for extending polymorphic constructions from free algebras to their quotients. The equivalence of projective algebras for a monad and all coalgebras for any of the corresponding comonads, claimed in the title of the paper, suggests a link between the problem of polymorphic quotients in monadic programming and the ideas of *comonadic* programming, put forward by several authors [17, 41, 2]. The dual link of injective algebras for a comonad and all algebras for any of the corresponding monads suggests a link between polymorphic quotients of cofree coalgebras and unrestricted quotients of algebras. We proceed to work out these links.

Prerequisites

This is a paper about categorical semantics of computation, so the prerequisites are mostly categorical. The main background definitions are reproduced in the Appendix. A very succinct overview of the underlying concepts can be found in [28, Sec. I.3].

1.2 Motivating example

Let \mathbb{C} be a cartesian closed category, i.e. given with an adjunction

$$\mathbb{C} \begin{array}{c} \xrightarrow{A \times} \\ \perp \\ \xleftarrow{A \Rightarrow} \end{array} \mathbb{C} \quad (1)$$

for every object A . Fix an object S as a *state space*, and consider the monad

$$\begin{aligned} \overleftarrow{S} : \mathbb{C} &\rightarrow \mathbb{C} \\ X &\mapsto S \Rightarrow (S \times X) \end{aligned} \quad (2)$$

induced by the adjunction (1) instantiated to S . As explained in [35], the category of free \overleftarrow{S} -algebras $\mathbb{C}_{\overleftarrow{S}}$ captures computations with explicit state, or with side effects. A computation over the inputs of type A , the outputs of type B , and the states of type S is presented as a free algebra homomorphism $f \in \mathbb{C}_{\overleftarrow{S}}(A, B)$, which can be conveniently viewed as a \mathbb{C} -morphism in the form $A \xrightarrow{f} S \Rightarrow (S \times B)$ [32]. This computation thus maps every input $a \in A$ to a function $S \xrightarrow{f(a)} S \times B$, determining at each state $s \in S$ a next state, and an output. Equivalently, such morphisms can be viewed in the transposed form $S \times A \xrightarrow{f'} S \times B$, assigning to each state and every input a next state and an output. This transposed form of homomorphisms between free algebras will turn out to be more convenient for the purposes of this paper. In the case of the state monad \overleftarrow{S} , such homomorphisms capture Mealy machines [15, 23, 26, Sec. 2.7(a)].

²The name is borrowed from theory of modules, where the retracts of free modules are also called projective.

Towards a more concrete example, consider the following model of data release policies from [39]. Suppose that a Mealy machine $S \times A \xrightarrow{f} S \times B$ models a database. S are its states, A are the inputs (insertions, queries, ...) that the users may enter, and B are the outputs supplied by the database. A stateless map $A \xrightarrow{g} B$ can be thought of a rudimentary deterministic channel, just mapping data of type A to data of type B . Since there are multiple users, there may be privacy policies, and authorization policies that need to be implemented. A privacy policy can be viewed as a map $S \times B \xrightarrow{\psi} S \times B$, which projects any output $b \in B$ of the database at a state $s \in S$ to a sanitized, public component $\psi(s, b)$ of the state and the output of the database, and filters out all private data. An authorization policy can be similarly viewed as a map $S \times A \xrightarrow{\varphi} S \times A$, which projects any database state $s \in S$, and any user input $a \in A$ (including the relevant credentials) to the authorized component $\varphi(s, a)$ of the state and the input. Since the public components should not contain any private residue to be filtered out in a second run of ψ , and the authorized components should not contain any unauthorized residue, the policies should be idempotent, i.e. satisfy the equations

$$\varphi \circ \varphi = \varphi \quad \psi \circ \psi = \psi$$

Such equations define projectors. There are at least two different ways to interpret projectors as policies. One is to view them as *policy specifications*. The database $S \times A \xrightarrow{f} S \times B$ then implements the policies $S \times B \xrightarrow{\psi} S \times B$ and $S \times A \xrightarrow{\varphi} S \times A$ if it satisfies the equation

$$f = \psi \circ f \circ \varphi \quad (3)$$

which is easily seen to be equivalent to the pair of equations

$$\psi \circ f = f = f \circ \varphi \quad (4)$$

In other words, a *compliant* database only ever supplies public data, and only ever permits authorized requests.

A different view to interpret projectors as policies is to view them as *policy implementations*. The database $S \times A \xrightarrow{f} S \times B$ then does not implement the policies itself, but needs to be precomposed with the authorization policy $S \times A \xrightarrow{\varphi} S \times A$ and postcomposed with the privacy policy $S \times B \xrightarrow{\psi} S \times B$. However, since each of these policies regulates the same data release, the database $S \times A \xrightarrow{f} S \times B$ is *consistent* with the policies $S \times B \xrightarrow{\psi} S \times B$ and $S \times A \xrightarrow{\varphi} S \times A$ if it satisfies the equation

$$\psi \circ f = f \circ \varphi \quad (5)$$

It is obvious that compliance implies consistency. A consistent database, however, does not have to be compliant. The reason is that a consistent database does not have to implement the policies itself, but it requires separate policy implementations at the input and at the output. On its own, such a database may accept unauthorized requests and it may supply private data. Its consistency means that *if* we make sure that no

unauthorized requests are submitted, *then* we can be sure that no private responses will be supplied, and *vice versa*. More precisely, a database f is consistent with the policies φ and ψ whenever a request consistent with φ results in a response consistent with ψ , and all responses consistent with ψ can be obtained on requests consistent with φ . Since the two policies thus precisely enforce each other on a database consistent with them, they implement the same data release process on this database, which can thus be implemented either as an authorization policy, or as a privacy policy.

Lifting this distinction between compliance and consistency from databases and state monads, to a distinction between two types of homomorphisms between projective algebras, we arrive at the main results of the paper.

1.3 The setting and the result

Every adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$ determines

- the monad $\overleftarrow{F} = F_* F^* : \mathbb{A} \rightarrow \mathbb{A}$, with the induced categories of
 - free algebras $\mathbb{A}_{\overleftarrow{F}}$,
 - projective algebras $\mathbb{A}_{\overleftarrow{F}}^\cup$, and
 - all algebras $\mathbb{A}^{\overleftarrow{F}}$,

and on the other hand

- the comonad $\overrightarrow{F} = F^* F_* : \mathbb{B} \rightarrow \mathbb{B}$, with the induced categories of
 - cofree coalgebras $\mathbb{B}_{\overrightarrow{F}}$,
 - injective coalgebras $\mathbb{B}_{\overrightarrow{F}}^\cup$, and
 - all coalgebras $\mathbb{B}^{\overrightarrow{F}}$.

The other way around, given a monad $\overleftarrow{F} : \mathbb{A} \rightarrow \mathbb{A}$, any adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$ such that $\overleftarrow{F} = F_* F^*$ is a *resolution* of \overleftarrow{F} ; and given a comonad $\overrightarrow{F} : \mathbb{B} \rightarrow \mathbb{B}$, any adjunction such that $\overrightarrow{F} = F^* F_*$ is a resolution of \overrightarrow{F} . Each of the above categories defined for a monad (resp. for a comonad) gives its resolution. The definitions are standard, and can be found in the Appendix. In this paper, we introduce the categories of

- projective algebras with *consistent* morphisms $\mathbb{A}_F^{\leftarrow}$, and
- injective coalgebras with *consistent* morphisms $\mathbb{B}_F^{\rightarrow}$.

We prove the following equivalences of categories

$$\mathbb{A}^{\overleftarrow{F}} \simeq \mathbb{A}_F^{\leftarrow} \qquad \mathbb{B}^{\overrightarrow{F}} \simeq \mathbb{B}_F^{\rightarrow} \tag{6}$$

under the assumption that the categories \mathbb{A} and \mathbb{B} are Cauchy³ complete, which means that they split idempotents. This is a very mild assumption, since the Cauchy completion, the weakest of all categorical completions, is easy to construct for any category, as spelled out in Prop. 2.2.

Overview of the paper

Sec. 2 begins with a discussion about projectors, and analyzes projectors over free algebras, which determine projective algebras. In Sec. 3 we state and prove the main theorem, showing that projectors over free algebras correspond to coalgebras. We also state the dual version, which says that projectors over cofree coalgebras correspond to algebras. In Sec. 4, we return to the motivating example from Sec. 1.2, and analyze the different categorical formalisations of data release policies. Sec. 5 closes the paper with comments about the related past work, and about the future work.

2 Projectors over algebras and coalgebras

2.1 Projectors in general

Consider an equalizer and coequalizer diagram

$$E \rightrightarrows^i A \begin{array}{c} \xrightarrow{\varphi} \\ \xrightarrow{\text{id}} \end{array} A \xrightarrow{q} \twoheadrightarrow Q$$

for an arbitrary endomorphism φ . Intuitively, the equalizer E consists of the *fixed points* of φ , whereas the coequalizer Q is the quotient where each element of A is identified with all of its direct and inverse images along φ , which together form its *orbit*. The obvious map $E \rightarrow Q$ maps each fixed point into a unique orbit; but some orbits may not contain any fixed points. We are interested in the situation when each orbit does contain a fixed point, so that each equivalence class from Q has a canonical representative in E . This means that the iterated applications of φ push each element of A along its orbit towards a fixed point. It can be shown that this situation is characterized by the requirement that the following countably extended diagram commutes

$$A \begin{array}{c} \xrightarrow{\varphi} \\ \xrightarrow{\text{id}} \end{array} A \xrightarrow{\varphi} A \xrightarrow{\varphi} \dots$$

In terms of elements, this means that for every $x \in A$ there is some $n \in \mathbb{N}$ such that $\varphi^{n+1}(x) = \varphi^n(x)$. In other words, φ thus equips A with the structure of a *forest*, where the equivalence classes that form Q are the component trees, and the elements of E are their roots. Projectors are the special case of this situation, where already the diagram

$$A \begin{array}{c} \xrightarrow{\varphi} \\ \xrightarrow{\text{id}} \end{array} A \xrightarrow{\varphi} A$$

³The habit of attributing categorical concepts to XIX century mathematicians and philosophers has styled the terminology in some parts of category theory.

commutes. The forest thus reduces to a shrub, where each component may branch at the root as wide as it likes, but can only grow one layer tall, and cannot grow any further branches. The following summarizes [7, Sec. IV.7.5].

Proposition 2.1. *If the endomorphism $A \xrightarrow{\varphi} A$ satisfies $\varphi \circ \varphi = \varphi$, then for any pair $A \begin{smallmatrix} \xrightarrow{q} \\ \xleftarrow{i} \end{smallmatrix} B$, the following statements are equivalent:*

(a) *i is an equalizer and q is a coequalizer of φ and id*

$$\begin{array}{ccc} & B & \\ i \swarrow & & \nwarrow q \\ A & \xrightarrow{\varphi} & A \\ \text{id} \xrightarrow{\quad} & & \end{array}$$

(b) *$i \circ q = \varphi$ and $q \circ i = \text{id}$.*

Definition 2.1. An endomorphism $A \xrightarrow{\varphi} A$ is called a *projector* (or *idempotent*) if $\varphi \circ \varphi = \varphi$. Its *splitting* is an object B with a pair of arrows $A \begin{smallmatrix} \xrightarrow{q} \\ \xleftarrow{i} \end{smallmatrix} B$ such that $i \circ q = \varphi$ and $q \circ i = \text{id}$. More succinctly, we write $A \xrightarrow{\varphi} B$.

Since the projectors and their splittings are defined by equations, every functor must preserve them. Since a splitting consists of an equalizer and a coequalizer, it is an equalizer and a coequalizer that must be preserved by all functors.

Definition 2.2. A categorical property is called *absolute* when it is preserved by all functors. A category that has all absolute limits and absolute colimits is said to be *Cauchy complete*.

It follows from the results of [36], as well as from the different approach in [16, Sec. I.6.5], that all absolute limits and colimits boil down to splittings.

Proposition 2.2. *For any category \mathbb{C} the following statements are equivalent*

(a) *\mathbb{C} is Cauchy complete,*

(b) *all projectors split in \mathbb{C} ,*

(c) *the obvious embedding $\mathbb{C} \hookrightarrow \mathbb{C}^\cup$ is an equivalence, where*

$$\begin{aligned} |\mathbb{C}^\cup| &= \coprod_{A \in |\mathbb{C}|} \{A \xrightarrow{\varphi} A \mid \varphi \circ \varphi = \varphi\} \\ \mathbb{C}^\cup(A \xrightarrow{\varphi} A, B \xrightarrow{\psi} B) &= \left\{ f \in \mathbb{C}(A, B) \mid \begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow \varphi & & \uparrow \psi \\ A & \xrightarrow{f} & B \end{array} \right\} \end{aligned}$$

The absolute completion \mathbb{C}^\cup is sometimes called *Karoubi envelope* of \mathbb{C} [7, Sec. IV.7.5]. Two categories are *Morita equivalent* when their Cauchy completions are equivalent [16, Thm. 7.9.4]. Note that the condition $\psi \circ f \circ \varphi = f$ is equivalent with the requirement that both $f \circ \varphi = f$ and $\psi \circ f = f$ are valid.

ASSUMPTION. In the rest of this paper, we assume that each of the categories under consideration is **Cauchy complete**, i.e. that projectors split in it. Any category \mathbb{C} that does not fulfill this assumption should be replaced by its Karoubi envelope \mathbb{C}^\cup , described in Prop. 2.2(c).

2.2 Projective algebras over free algebras

In homological algebra, projective modules are usually defined as direct summands of free modules. In the terminology of the preceding section, this means that they arise by *splitting the projectors* over free modules. It is natural to define projective algebras in a similar way: as projectors over free algebras [22, §82].

In the usual (Kleisli) view of the category of free algebras [32], reproduced in the Appendix, a morphism $f \in \mathbb{A}_{\overleftarrow{F}}(A, B)$ is a morphism $A \xrightarrow{f} \overleftarrow{F}B$ in \mathbb{A} , and its composition with $g \in \mathbb{A}_{\overleftarrow{F}}(B, C)$, which is $B \xrightarrow{g} \overleftarrow{F}C$ in \mathbb{A} is defined by

$$g \circ_{\overleftarrow{F}} f = \left(A \xrightarrow{f} \overleftarrow{F}B \xrightarrow{\overleftarrow{F}g} \overleftarrow{F}\overleftarrow{F}C \xrightarrow{\mu} \overleftarrow{F}C \right) \in \mathbb{A}_{\overleftarrow{F}}(A, C)$$

A projector $\varphi \in \mathbb{A}_{\overleftarrow{F}}(A, A)$ over the free algebra generated by A is thus an \mathbb{A} -morphism $A \xrightarrow{\varphi} \overleftarrow{F}A$ such that $\varphi \circ_{\overleftarrow{F}} \varphi = \mu_A \circ \overleftarrow{F}\varphi \circ \varphi = \varphi$. As they expand, the calculations with projectors in the Kleisli form of category $\mathbb{A}_{\overleftarrow{F}}$ do get increasingly clumsy.

When a monad $\overleftarrow{F} : \mathbb{A} \rightarrow \mathbb{A}$ is induced by an adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$ so that $\overleftarrow{F} = F_*F^*$, then the category of free algebras can be equivalently defined by

$$\begin{aligned} |\mathbb{A}_{\overleftarrow{F}}| &= |\mathbb{A}| \\ \mathbb{A}_{\overleftarrow{F}}(X, Y) &= \mathbb{B}(F^*X, F^*Y) \end{aligned}$$

It is easy to check that the natural bijections

$$\mathbb{A}(X, F_*F^*Y) \cong \mathbb{B}(F^*X, F^*Y)$$

map the Kleisli composition of the morphisms in $\mathbb{A}(X, \overleftarrow{F}Y)$ to the ordinary composition of their adjunction transposes in $\mathbb{B}(F^*X, F^*Y)$. If the homomorphisms between free \overleftarrow{F} -algebras are presented as the elements of $\mathbb{B}(F^*X, F^*Y)$, then a projector $\varphi \in \mathbb{A}_{\overleftarrow{F}}(X, X)$ is just a \mathbb{B} -morphism $F^*X \xrightarrow{\varphi} F^*X$ such that $\varphi \circ \varphi = \varphi$. The category of projective \overleftarrow{F} -algebras and *compliant* homomorphisms (explained in Sec. 1.2) is thus defined as

follows:

$$|\mathbb{A}_{\overleftarrow{F}}^\cup| = \coprod_{X \in |\mathbb{A}|} \{ \varphi \in \mathbb{B}(F^*X, F^*X) \mid \varphi \circ \varphi = \text{id} \}$$

$$\mathbb{A}_{\overleftarrow{F}}^\cup(\varphi, \psi) = \left\{ h \in \mathbb{B}(F^*X, F^*Y) \mid \begin{array}{ccc} F^*X & \xrightarrow{h} & F^*Y \\ \downarrow \varphi & & \uparrow \psi \\ F^*X & \xrightarrow{h} & F^*Y \end{array} \right\}$$

where $\psi \in \mathbb{B}(F^*C, F^*C)$ is another projective algebra, viewed as a projector over the free algebra generated by $C \in |\mathbb{A}|$.

2.3 Projective algebras among all algebras

The category of free \overleftarrow{F} -algebras $\mathbb{A}_{\overleftarrow{F}}$ embeds fully and faithfully into the category $\mathbb{A}^{\overleftarrow{F}}$ of all \overleftarrow{F} -algebras by the functor

$$\mathbb{A}_{\overleftarrow{F}} \xhookrightarrow{M} \mathbb{A}^{\overleftarrow{F}} \quad (7)$$

defined by

$$\frac{X \in |\mathbb{A}_{\overleftarrow{F}}|}{(\overleftarrow{F}\overleftarrow{F}X \xrightarrow{\mu} \overleftarrow{F}X) \in |\mathbb{A}^{\overleftarrow{F}}|} \quad \text{and} \quad \frac{(F^*X \xrightarrow{h} F^*Y) \in \mathbb{A}_{\overleftarrow{F}}(X, Y)}{(\overleftarrow{F}X \xrightarrow{F_*h} \overleftarrow{F}Y) \in \mathbb{A}^{\overleftarrow{F}}(\mu_X, \mu_Y)}$$

where F_*h is an algebra homomorphism because $\mu = F_*\varepsilon$, and the naturality of ε thus implies $F_*h \circ \mu_X = \mu_Y \circ \overleftarrow{F}F_*h$. Since the projectors in $\mathbb{A}^{\overleftarrow{F}}$ split whenever they split in \mathbb{A} , as assumed here, the embedding M has a unique extension M^\cup ,

$$\begin{array}{ccc} \mathbb{A}_{\overleftarrow{F}}^\cup & \xrightarrow{M^\cup} & \mathbb{A}^{\overleftarrow{F}} \\ \downarrow & \nearrow M & \\ \mathbb{A}_{\overleftarrow{F}} & & \end{array} \quad (8)$$

which maps each $\varphi \in |\mathbb{A}_{\overleftarrow{F}}^\cup|$ to a splitting α of the projector $F_*\varphi \in \mathbb{A}^{\overleftarrow{F}}(\mu_X, \mu_X)$

$$\begin{array}{ccccc} \overleftarrow{F}\overleftarrow{F}X & \xrightarrow{\overleftarrow{F}q} & \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}i} & \overleftarrow{F}\overleftarrow{F}X \\ \mu \downarrow & & \downarrow \alpha & & \downarrow \mu \\ \overleftarrow{F}X & \xrightarrow{q} & A & \xrightarrow{i} & \overleftarrow{F}X \\ & \searrow F_*\varphi & & & \end{array} \quad (9)$$

An \overleftarrow{F} -algebra $\overleftarrow{F}A \xrightarrow{\alpha} A$ is thus projective if it is a retract of some free algebra $\overleftarrow{F}\overleftarrow{F}X \xrightarrow{\mu} \overleftarrow{F}X$, i.e. if there are an $X \in |\mathbb{A}|$ and some homomorphisms $q \in \mathbb{A}^{\overleftarrow{F}}(\mu_X, \alpha)$ and $i \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_X)$ such that $q \circ i = \text{id}$.

It turns out, however, that each projective algebra $\overleftarrow{F}A \xrightarrow{\alpha} A$ is not just a retract of a free algebra over some $X \in |\mathbb{A}|$, but a retract of the free algebra over its own carrier A .

Proposition 2.3. *An algebra $\overleftarrow{F}A \xrightarrow{\alpha} A$ is projective if and only if there is a unique algebra homomorphism $\overline{\alpha} \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_A)$ such that $\alpha \circ \overline{\alpha} = \text{id}$.*

$$\begin{array}{ccccc} \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}\overline{\alpha}} & \overleftarrow{F}\overleftarrow{F}A & \xrightarrow{\overleftarrow{F}\alpha} & \overleftarrow{F}A \\ \alpha \downarrow & & \downarrow \mu_A & & \downarrow \alpha \\ A & \xrightarrow{\overline{\alpha}} & \overleftarrow{F}A & \xrightarrow{\alpha} & A \end{array}$$

Proof. To construct $\overline{\alpha} \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_A)$, extend the algebra homomorphism $q \in \mathbb{A}^{\overleftarrow{F}}(\mu_X, \alpha)$ to $\overleftarrow{F}(q \circ \eta) \in \mathbb{A}^{\overleftarrow{F}}(\mu_X, \mu_A)$ and precompose with $i \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_X)$. Hence $\overline{\alpha} = \overleftarrow{F}q \circ \overleftarrow{F}\eta \circ i \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_A)$, as displayed in the middle row of the following diagram.

$$\begin{array}{ccccccc} \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}i} & \overleftarrow{F}\overleftarrow{F}X & \xrightarrow{\overleftarrow{F}\overleftarrow{F}\eta} & \overleftarrow{F}\overleftarrow{F}\overleftarrow{F}X & \xrightarrow{\overleftarrow{F}\overleftarrow{F}q} & \overleftarrow{F}\overleftarrow{F}A \\ \alpha \downarrow & & \downarrow \mu & & \downarrow \mu & & \downarrow \mu \\ A & \xrightarrow{i} & \overleftarrow{F}X & \xrightarrow{\overleftarrow{F}\eta} & \overleftarrow{F}\overleftarrow{F}X & \xrightarrow{\overleftarrow{F}q} & \overleftarrow{F}A \\ & & \searrow \text{id} & & \downarrow \mu & & \downarrow \alpha \\ & & & & \overleftarrow{F}X & \xrightarrow{q} & A \end{array}$$

The commutativity of the upper three squares implies that $\overline{\alpha}$ is an \overleftarrow{F} -algebra homomorphism. The commutativity of the lower square and the triangle implies that $\alpha \circ \overline{\alpha} = q \circ i = \text{id}$.

To see that $\overline{\alpha}$ is unique, i.e. that it is the only way to display $\overleftarrow{F}A \xrightarrow{\alpha} A$ as a subalgebra of $\overleftarrow{F}\overleftarrow{F}A \xrightarrow{\mu} \overleftarrow{F}A$, note that the composite homomorphism $\overline{\alpha} \circ \alpha$ is a projector on the free algebra $\overleftarrow{F}\overleftarrow{F}A \xrightarrow{\mu} \overleftarrow{F}A$, and that $\overleftarrow{F}A \xrightarrow{\alpha} A$ is the splitting of this projector.

$$\begin{array}{ccccc} \overleftarrow{F}\overleftarrow{F}A & \xrightarrow{\overleftarrow{F}\alpha} & \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}\overline{\alpha}} & \overleftarrow{F}\overleftarrow{F}A \\ \mu \downarrow & & \alpha \downarrow & & \downarrow \mu \\ \overleftarrow{F}A & \xrightarrow{\alpha} & A & \xrightarrow{\overline{\alpha}} & \overleftarrow{F}A \end{array}$$

While the splitting of a projector is unique up to an isomorphism, fixing one component of the splitting determines the other one on-the-nose: since α is an epi, $\overline{\alpha}_0 \circ \alpha = \overline{\alpha}_1 \circ \alpha$ implies that $\overline{\alpha}_0 = \overline{\alpha}_1$. \square

The preceding proposition thus says that every projective algebra $\overleftarrow{F}A \xrightarrow{\alpha} A$ has a unique embedding $A \xrightarrow{\overline{\alpha}} \overleftarrow{F}A$ into the free algebra μ_A over its carrier A . With no loss

of generality, the full subcategory $\mathbb{A}_{\cup}^{\overleftarrow{F}}$ of the (Eilenberg-Moore) category $A^{\overleftarrow{F}}$ of all \overleftarrow{F} -algebras spanned by the projective \overleftarrow{F} -algebras can thus be viewed in the form

$$\begin{aligned} |\mathbb{A}_{\cup}^{\overleftarrow{F}}| &= \left\{ \left(\overleftarrow{F}A \xrightarrow{\alpha} A \right) \in |\mathbb{A}^{\overleftarrow{F}}| \mid \exists \overline{\alpha} \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_A). \alpha \circ \overline{\alpha} = \text{id}_A \right\} \\ &\cong \coprod_{\alpha \in |\mathbb{A}^{\overleftarrow{F}}|} \left\{ \overline{\alpha} \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \mu_A) \mid \alpha \circ \overline{\alpha} = \text{id}_A \right\} \\ \mathbb{A}_{\cup}^{\overleftarrow{F}}(\alpha, \gamma) &= \left\{ f \in \mathbb{A}(A, C) \mid \begin{array}{ccc} \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}f} & \overleftarrow{F}C \\ \downarrow \alpha & & \downarrow \gamma \\ A & \xrightarrow{f} & C \end{array} \right\} \end{aligned}$$

Lemma 2.4. *Let $\overleftarrow{F}A \xrightarrow{\alpha} A$ be a projective \overleftarrow{F} -algebra and $A \xrightarrow{\overline{\alpha}} \overleftarrow{F}A$ the \overleftarrow{F} -algebra monomorphism, as constructed in Prop.2.3, that makes α into a subalgebra of the free algebra μ_A . Then the transpose $F^*A \xrightarrow{\overline{\alpha}'} F^*A$ of $A \xrightarrow{\overline{\alpha}} \overleftarrow{F}A$ is idempotent⁴, and thus a projector in $\mathbb{A}_{\overleftarrow{F}}$. Moreover*

$$\overline{\alpha} \circ \alpha = F_* \overline{\alpha}' \quad (10)$$

Proof. The fact that $\overline{\alpha}' \circ \overline{\alpha}' = \overline{\alpha}'$ can be seen by transposing the following diagram along the adjunction $F^* \dashv F_*$.

$$\begin{array}{ccccc} A & \xrightarrow{\overline{\alpha}} & \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}\overline{\alpha}} & \overleftarrow{F}\overleftarrow{F}A \\ & \searrow \text{id} & \downarrow \alpha & & \downarrow \mu \\ & & A & \xrightarrow{\overline{\alpha}} & \overleftarrow{F}A \end{array}$$

Equation (10) also follows from the commutativity of the square on the above diagram, and the observation that

$$\mu \circ \overleftarrow{F}\overline{\alpha} = F_* \varepsilon \circ F_* F^* \overline{\alpha} = F_* (\varepsilon \circ F^* \overline{\alpha}) = F_* \overline{\alpha}'$$

□

Lemma 2.5. *Let $\overleftarrow{F}A \xrightarrow{\alpha} A$ and $\overleftarrow{F}C \xrightarrow{\gamma} C$ be projective \overleftarrow{F} -algebras, with the \overleftarrow{F} -algebra monomorphisms $A \xrightarrow{\overline{\alpha}} \overleftarrow{F}A$ and $C \xrightarrow{\overline{\gamma}} \overleftarrow{F}C$ including them into the free \overleftarrow{F} -algebras μ_A and μ_C , respectively, as in Prop. 2.3. Then every \overleftarrow{F} -algebra homomorphism $f \in \mathbb{A}_{\cup}^{\overleftarrow{F}}(\alpha, \gamma) = \mathbb{A}_{\cup}^{\overleftarrow{F}}(\alpha, \gamma)$ induces the homomorphism $Hf = \overline{\gamma}' \circ F^*f = F^*f \circ \overline{\alpha}' \in \mathbb{A}_{\overleftarrow{F}}^{\cup}(\overline{\alpha}', \overline{\gamma}')$, which is compliant in the sense of (3), since*

$$f \circ \alpha = \gamma \circ \overleftarrow{F}f \iff \overline{\gamma}' \circ Hf \circ \overline{\alpha}' = Hf$$

⁴If the category of free algebras $\mathbb{A}_{\overleftarrow{F}}$ is presented in the Kleisli form, then $A \xrightarrow{\overline{\alpha}} \overleftarrow{F}A$ itself is an idempotent morphism, and thus a projector in it.

Proposition 2.6. *There is an equivalence of categories*

$$\mathbb{A}_{\cup}^{\overleftarrow{F}} \begin{array}{c} \xrightarrow{H} \\ \xrightarrow[\approx]{K} \end{array} \mathbb{A}_{\overline{F}}^{\cup} \quad (11)$$

where the object part of the functor H is defined using Lemma 2.4

$$\frac{(\alpha \xrightarrow{\overline{\alpha}} \mu_A) \in |\mathbb{A}_{\cup}^{\overleftarrow{F}}|}{H\alpha = \langle A, F^*A \xrightarrow{\overline{\alpha}'} F^*A \rangle \in |\mathbb{A}_{\overline{F}}^{\cup}|}$$

whereas the arrow part is defined in Lemma 2.5.

$$\frac{f \in \mathbb{A}_{\cup}^{\overleftarrow{F}}(\alpha, \gamma)}{Hf = \overline{\gamma}' \circ F^*f = F^*f \circ \overline{\alpha}' \in \mathbb{A}_{\overline{F}}^{\cup}(H\alpha, H\gamma)}$$

The functor K , on the other hand, is the factorization of the functor $M^{\cup} : \mathbb{A}_{\overline{F}}^{\cup} \rightarrow \mathbb{A}_{\cup}^{\overleftarrow{F}}$ from (8) through the inclusion $\mathbb{A}_{\cup}^{\overleftarrow{F}} \hookrightarrow \mathbb{A}_{\overline{F}}^{\cup}$. Its object and the arrow parts

$$\frac{\langle X, F^*X \xrightarrow{\varphi} F^*X \rangle \in |\mathbb{A}_{\overline{F}}^{\cup}|}{(\overleftarrow{F}A \xrightarrow{K\varphi} A) \in |\mathbb{A}_{\cup}^{\overleftarrow{F}}|} \quad \text{and} \quad \frac{h \in \mathbb{A}_{\overline{F}}^{\cup}(\varphi, \psi)}{Kh \in \mathbb{A}_{\cup}^{\overleftarrow{F}}(K\varphi, K\psi)}$$

are defined by the projector splittings

$$\begin{array}{ccc} \textcircled{F_*\varphi} & \mu_X & \begin{array}{c} \xrightarrow{q} \\ \xrightarrow[i]{K\varphi} \end{array} \\ \downarrow F_*h & & \downarrow Kh \\ \textcircled{F_*\psi} & \mu_Y & \begin{array}{c} \xrightarrow{p} \\ \xrightarrow[j]{K\psi} \end{array} \end{array} \quad (12)$$

Proof. Both functors are clearly well defined. Towards the isomorphism $HK\varphi \cong \varphi$, we first split the projector $\mu_X \xrightarrow{F_*\varphi} \mu_X$ in $\mathbb{A}_{\cup}^{\overleftarrow{F}}$ to get the subalgebra $K\varphi = (\overleftarrow{F}A \xrightarrow{\alpha} A)$ of the free algebra $\overleftarrow{F}F^*X \xrightarrow{\mu} \overleftarrow{F}X$, and then construct the homomorphism $A \xrightarrow{\overline{\alpha}} \overleftarrow{F}A$, like in Prop. 2.3, to get the projector $HK\varphi = H\alpha = \overline{\alpha}'$. The isomorphism $\varphi \cong HK\varphi$ in $\mathbb{A}_{\overline{F}}^{\cup}$ is realized by the transpose $F^*A \xrightarrow{i'} F^*X$ of the monic component $A \xrightarrow{i} F_*F^*X$ of the splitting $F_*\varphi = (F^*F_*X \xrightarrow{q} A \xrightarrow{i} F^*F_*X)$ in (9). The fact that i' is a projector homomorphism from $\overline{\alpha}' = HK\varphi$ to φ in $\mathbb{A}_{\overline{F}}^{\cup}$ boils down to the equations $\varphi \circ i' = i' = i' \circ \overline{\alpha}'$. To see that the first one holds, take a look at the adjunction transposes of its two sides:

$$F_*\varphi \circ i = i \circ q \circ i = i$$

To see that the second equation holds, consider the following diagram:

$$\begin{array}{ccccc}
 & & \overline{\alpha} & & \\
 A & \xrightarrow{i} & \overleftarrow{F}X & \xrightarrow{\eta} & \overleftarrow{F}\overleftarrow{F}X & \xrightarrow{\overleftarrow{F}q} & \overleftarrow{F}A \\
 & \searrow i & \downarrow q & & \downarrow \overleftarrow{F}F_*\varphi & & \downarrow \overleftarrow{F}i \\
 & & A & \xrightarrow{F_*\varphi} & \overleftarrow{F}A & & \\
 & & \downarrow i & & \downarrow \overleftarrow{F}i & & \\
 & & \overleftarrow{F}X & \xrightarrow{\eta} & \overleftarrow{F}\overleftarrow{F}X & & \\
 & & \downarrow \mu & & & &
 \end{array}$$

The two paths around this diagram correspond to the transposes of the two sides of the second equation. Hence $i' \in \mathbb{A}_{\overleftarrow{F}}^{\cup}(\overline{\alpha}', \varphi)$. To show that i' is an isomorphism in $\mathbb{A}_{\overleftarrow{F}}^{\cup}$, consider

$$i'' = (F^*X \xrightarrow{F^*\eta} F^*F_*F^*X \xrightarrow{F^*q} F^*A \xrightarrow{\overline{\alpha}'} F^*A) \quad (13)$$

The equations

$$i'' \circ i' = \overline{\alpha}' \quad \text{and} \quad i' \circ i'' = \varphi \quad (14)$$

follow directly from the definitions. Since this immediately implies $i'' \circ \varphi = i'' \circ \overline{\alpha}' \circ i''$, it follows that $i'' \in \mathbb{A}_{\overleftarrow{F}}^{\cup}(\varphi, \overline{\alpha}')$. Equations (14) mean that i' and i'' are each other's inverses in $\mathbb{A}_{\overleftarrow{F}}^{\cup}$. Hence $HK\varphi \cong \varphi$, where $HK\varphi = \overline{\alpha}'$.

The isomorphism $KH\alpha \cong \alpha$ may seem surprising. How can the functor $Hf = \overline{\gamma}' \circ F^*f = F^*f \circ \overline{\alpha}'$ be faithful when F^* in general does not have to be faithful? The answer is in the following diagram:

$$\begin{array}{ccccc}
 & & F_*Hf = F_*\overline{\gamma}' \circ F_*F^*f & & \\
 \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}f} & \overleftarrow{F}C & \xrightarrow{\gamma} & C & \xrightarrow{\overline{\gamma}} & \overleftarrow{F}C \\
 \downarrow q & & \downarrow \gamma & & \downarrow \gamma & & \downarrow \gamma \\
 A & \xrightarrow{f} & C & \xrightarrow{\text{id}} & C & & C \\
 \downarrow \overline{\alpha} & & \downarrow \overline{\gamma} & & \downarrow \overline{\gamma} & & \downarrow \overline{\gamma} \\
 \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}f} & \overleftarrow{F}C & \xrightarrow{\gamma} & C & \xrightarrow{\overline{\gamma}} & \overleftarrow{F}C \\
 & & F_*Hf = F_*\overline{\gamma}' \circ F_*F^*f & &
 \end{array}$$

On the left and on the right are the projectors $F_*H\alpha$ and $F_*H\gamma$. By (12), splitting them gives $KH\alpha \cong \alpha$ and $KH\gamma \cong \gamma$. On the top and on the bottom is the projector morphism F_*Hf . Also by (12), it induces the morphism $Khf \in \mathbb{A}_{\overleftarrow{F}}^{\cup}(\alpha, \gamma)$ between the splittings of $F_*H\alpha$ and $F_*H\gamma$. It is denoted by the dashed arrow through the middle. We show that $Khf = f$. Since the projector splittings are given as the epi-mono factorizations, Khf is the unique morphism from A on the left to C on the right making the rectangle

above it and the rectangle below it commute. But the vertical arrow $\overleftarrow{F}C \rightarrow C \rightarrow \overleftarrow{F}C$ is also an epi-mono factorization, and

- f is the unique morphism making the upper left rectangle and the lower left rectangle commute (the latter by Lemma 2.5);
- id is the unique morphism making the upper right rectangle and the lower right rectangle commute (because $\gamma \circ \overline{\gamma} = \text{id}$).

Hence $KHf = f$. □

3 Equivalences between algebras and coalgebras

In this Section we prove the main theorem, establishing the equivalence between projective algebras and all coalgebras, and state the dual theorem, establishing the equivalence between injective coalgebras and all algebras. The equivalences, however, require the *consistent* homomorphisms, as in (5), and not the compliant homomorphisms, like in (3) and $\mathbb{A}_{\overleftarrow{F}}^{\cup}$.

3.1 Consistent homomorphisms

We define the category of projective \overleftarrow{F} -algebras and consistent homomorphisms in two forms, one over the projectors in $\mathbb{A}_{\overleftarrow{F}}$, one as a subcategory of $\mathbb{A}_{\overleftarrow{F}}$ again. The first version is:

$$\begin{aligned} |\mathbb{A}_{\overleftarrow{F}}^{\leftarrow \epsilon}| &= \coprod_{X \in |\mathbb{A}|} \{ \varphi \in \mathbb{B}(F^*X, F^*X) \mid \varphi \circ \varphi = \varphi \wedge \overleftarrow{F}X \xrightarrow{F_*\varphi} X \} \\ \mathbb{A}_{\overleftarrow{F}}^{\leftarrow \epsilon}(\varphi, \psi) &= \left\{ f \in \mathbb{A}(X, Y) \mid \begin{array}{ccc} F^*X & \xrightarrow{F^*f} & F^*Y \\ \downarrow \varphi & & \downarrow \psi \\ F^*X & \xrightarrow{F^*f} & F^*Y \end{array} \right\} \end{aligned}$$

where $\overleftarrow{F}X \xrightarrow{F_*\varphi} X$ is the notation from Def. 2.1, meaning that $F_*\varphi$ splits in the form $\overleftarrow{F}X \rightarrow X \rightarrow \overleftarrow{F}X$. The second version is:

$$\begin{aligned} |\mathbb{A}_{\leftarrow \epsilon}^F| &= \coprod_{\alpha \in |\mathbb{A}_{\overleftarrow{F}}|} \{ \overline{\alpha} \in \mathbb{A}_{\overleftarrow{F}}^{\leftarrow}(\alpha, \mu_A) \mid \alpha \circ \overline{\alpha} = \text{id}_A \} \\ \mathbb{A}_{\leftarrow \epsilon}^F(\alpha, \gamma) &= \left\{ f \in \mathbb{A}_{\overleftarrow{F}}^{\leftarrow}(\alpha, \gamma) \mid \begin{array}{ccc} \overleftarrow{F}A & \xrightarrow{\overleftarrow{F}f} & \overleftarrow{F}C \\ \uparrow \overline{\alpha} & & \uparrow \overline{\gamma} \\ A & \xrightarrow{f} & C \end{array} \right\} \end{aligned}$$

Remark. In \mathbb{A}_F^\cup we did not require that every projector $F^*X \xrightarrow{\varphi} F^*X$ satisfies $\overleftarrow{F}X \xrightarrow{F_*\varphi} X$, as in the definition of $\mathbb{A}_F^{\leftarrow\varphi}$ above. The reason is that in \mathbb{A}_F^\cup , every projector $F^*X \xrightarrow{\varphi} F^*X$ is isomorphic to a projector $F^*A \xrightarrow{\overline{\alpha}'} F^*A$, induced by a projective \overleftarrow{F} -algebra $\overleftarrow{F}A \xrightarrow{\alpha} A$. This latter projector always satisfies the requirement $\overleftarrow{F}A \xrightarrow{F_*\overline{\alpha}'} A$, because $F_*\alpha'$ splits into $\overleftarrow{F}A \xrightarrow{\alpha} A \xrightarrow{\overline{\alpha}'} \overleftarrow{F}A$, as proved in Lemma 2.4. This isomorphism $\varphi \cong \overline{\alpha}'$ was spelled out in the proof of Prop. 2.6, leading to the natural isomorphism $\varphi \cong HK\varphi$. However, this isomorphism is generally not consistent: it is present in \mathbb{A}_F^\cup , but not in $\mathbb{A}_F^{\leftarrow\varphi}$. This is why the requirement $\overleftarrow{F}A \xrightarrow{F_*\varphi} A$ needs to be explicitly imposed on the objects of $\mathbb{A}_F^{\leftarrow\varphi}$, if the equivalence $\mathbb{A}_F^{\leftarrow\varphi} \simeq \mathbb{A}_F^\cup$ from Prop. 2.6 is to be extended to $\mathbb{A}_F^{\leftarrow\varphi} \simeq \mathbb{A}_F^{\leftarrow\varphi}$.

Proposition 3.1. *There is an equivalence of categories*

$$\mathbb{A}_F^F \begin{array}{c} \xleftarrow{H} \\ \xrightarrow{K} \end{array} \mathbb{A}_F^{\leftarrow\varphi}$$

where the object parts of both functors are as defined in Prop. 2.6, whereas the arrow parts are

$$\frac{f \in \mathbb{A}_F^F(\alpha, \gamma)}{Hf = f \in \mathbb{A}_F^{\leftarrow\varphi}(\overline{\alpha}', \overline{\gamma}')} \quad \text{and} \quad \frac{h \in \mathbb{A}_F^{\leftarrow\varphi}(\varphi, \psi)}{Kh \in \mathbb{A}_F^F(K\varphi, K\psi)}$$

Proof. We begin like in the proof of Prop. 2.6: towards the isomorphism $HK\varphi \cong \varphi$, we first split the projector $\mu_X \xrightarrow{F_*\varphi} \mu_X$ in \mathbb{A}_F^F to $\mu_X \xrightarrow{\alpha} \alpha \xrightarrow{\overline{\alpha}'} \mu_X$. This time, however, the assumption $\overleftarrow{F}A \xrightarrow{F_*\overline{\alpha}'} A$ means that $F_*\varphi$ has a splitting in the form $\overleftarrow{F}X \xrightarrow{q} X \xrightarrow{i} \overleftarrow{F}X$. The carrier of the algebra α must be isomorphic to X , and can be chosen to be X itself. Since μ_X is a free algebra, it has a unique \overleftarrow{F} -algebra homomorphism to α . Since both α and q are homomorphisms $\mu_X \rightarrow \alpha$, it follows that $q = \alpha$. Since each component of a splitting determines the other one, it follows that $i = \overline{\alpha}$. Hence $F_*\varphi = \overline{\alpha} \circ \alpha$. It follows that $\varphi = \overline{\alpha}'$, because

$$\varphi' = F_*\varphi \circ \eta = \overline{\alpha} \circ \alpha \circ \eta = \overline{\alpha}$$

Thus $HK\varphi = \overline{\alpha}' = \varphi$. The natural isomorphism $KH\alpha \cong \alpha$ is constructed like in Prop. 2.6. The only additional observation is that the condition defining the consistent morphisms in $\mathbb{A}_F^{\leftarrow\varphi}$ and the condition defining the inclusion preserving \overleftarrow{F} -algebra homomorphisms in \mathbb{A}_F^F are each other's adjunction tranpose. \square

3.2 Projective algebras as coalgebras

Theorem 3.2. *For every adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$, with the induced monad $\overleftarrow{F} = F_*F^*$ and comonad $\overrightarrow{F} = F^*F_*$, the category of \overrightarrow{F} -coalgebras is equivalent with the category of projective \overleftarrow{F} -algebras and consistent homomorphisms, provided that \mathbb{B}*

is Cauchy complete. The equivalence is given by the functors

$$\mathbb{B}^{\vec{F}} \begin{array}{c} \xrightarrow{R} \\ \xleftarrow{L} \end{array} \mathbb{A}_F^{\leftarrow \circ}$$

where the rules

$$\frac{(B \xrightarrow{\beta} \vec{F}B) \in |\mathbb{B}^{\vec{F}}|}{R\beta = \langle F_*B, \vec{F}B \xrightarrow{\varepsilon} B \xrightarrow{\beta} \vec{F}B \rangle \in |\mathbb{A}_F^{\leftarrow \circ}|}$$

and

$$\frac{(B \xrightarrow{g} D) \in \mathbb{B}^{\vec{F}}(\beta, \delta)}{Rg = \left(\vec{F}B \xrightarrow{\vec{F}g} \vec{F}D \right) \in \mathbb{A}_F^{\leftarrow \circ}(R\beta, R\delta)}$$

define R , whereas the object part of L

$$\frac{\langle A, F^*A \xrightarrow{\varphi} F^*A \rangle \in |\mathbb{A}_F^{\leftarrow \circ}|}{L\varphi = \left(B \xrightarrow{i} F^*A \xrightarrow{F^*q'} F^*F_*B \right) \in |\mathbb{B}^{\vec{F}}|}$$

and its arrow part

$$\frac{(A \xrightarrow{f} C) \in \mathbb{A}_F^{\leftarrow \circ}(\varphi, \psi)}{(B \xrightarrow{Lf} D) \in \mathbb{B}^{\vec{F}}(L\varphi, L\psi)}$$

are defined using the projector splittings in the following diagram

$$\begin{array}{ccccc} F^*F_*B & \xrightarrow{\sim} & F^*A & \xrightarrow{q} & B & \xrightarrow{i} & F^*A & \xrightarrow{\sim} & F^*F_*B \\ & \downarrow F^*F_*Lf & \downarrow F^*f & \downarrow Lf & \downarrow F^*f & \downarrow F^*F_*Lf & & & \\ F^*F_*D & \xrightarrow{\sim} & F^*C & \xrightarrow{p} & D & \xrightarrow{j} & F^*C & \xrightarrow{\sim} & F^*F_*D \end{array} \quad (15)$$

$\begin{array}{c} \varphi \\ \curvearrowright \\ \psi \end{array}$
 $\begin{array}{c} L\varphi \\ \curvearrowright \\ L\psi \end{array}$

Proof. The functor R is well defined, i.e. it lands in $\mathbb{A}_F^{\leftarrow \circ}$, because $\varepsilon \circ \beta = \text{id}_B$, which implies that $R\beta$ is a projector:

$$R\beta \circ R\beta = \beta \circ \varepsilon \circ \beta \circ \varepsilon = \beta \circ \varepsilon = R\beta$$

The fact that $Rg = \left(\vec{F}B \xrightarrow{\vec{F}g} \vec{F}D \right)$ is a consistent $\mathbb{A}_F^{\leftarrow \circ}$ -morphism follows from the naturality of ε and the fact that $B \xrightarrow{g} D$ is an \vec{F} -coalgebra homomorphism.

To show that the functor L is well defined, we need to prove that $L\varphi = F^*q' \circ i$ is an \vec{F} -coalgebra, and that Lf , as defined in (15), is an \vec{F} -coalgebra homomorphism. The former requirement means that $L\varphi$ must satisfy the coalgebra equations:

$$\varepsilon \circ L\varphi = \text{id}_B \quad (16)$$

$$F^*F_*L\varphi \circ L\varphi = F^*\eta \circ L\varphi \quad (17)$$

To spell this out, consider diagram (15). The object B is defined by the splitting $F^*A \xrightarrow{q} B \xrightarrow{i} F^*A$ of the projector φ ; the object D is defined by the splitting $F^*C \xrightarrow{p} D \xrightarrow{j} F^*C$ of the projector ψ . On the other hand, using the equivalence of categories $\mathbb{A}_{\leftarrow}^F \xrightarrow{H} \mathbb{A}_F^{\leftarrow}$ from Prop. 3.1, we can assume without loss of generality that $\varphi = H\alpha = \bar{\alpha}'$ and $\psi = H\gamma = \bar{\gamma}'$ for some projective algebras $\alpha, \gamma \in |\mathbb{A}_{\leftarrow}^F|$, where the inclusions $\alpha \xrightarrow{\bar{\alpha}} \mu_A$ and $\gamma \xrightarrow{\bar{\gamma}} \mu_C$ transpose to $F^*A \xrightarrow{\bar{\alpha}'} F^*A$ and $F^*C \xrightarrow{\bar{\gamma}'} F^*C$. Lemma 2.4 says that the projector $F_*\varphi = F_*\bar{\alpha}'$ splits into $F_*F^*A \xrightarrow{\bar{\alpha}} A \xrightarrow{\alpha} F_*F^*A$. Since every functor preserves projector splittings, the F_* -image of the splitting $F^*A \xrightarrow{q} B \xrightarrow{i} F^*A$ of φ is also a splitting of $F_*\varphi$. The two splittings of $F_*\varphi$ induce an isomorphism $A \cong F_*B$. By chasing the following diagram, we see that this isomorphism is the transpose $A \xrightarrow{q'} F_*B$ of $F^*A \xrightarrow{q} B$.

$$\begin{array}{ccccc} & & F_*B & & \\ & \nearrow^{F_*q} & \uparrow & \nwarrow_{F_*i} & \\ F_*F^*A & & & & F_*F^*A \\ & \nwarrow_{\eta} & \downarrow q' & \nearrow_{\bar{\alpha}} & \\ & & A & & \end{array} \quad (18)$$

If we define $F_*B \xrightarrow{q''} A_*$ to be the inverse of the isomorphism $A \xrightarrow{q'} F_*B$, then the transposition gives

$$\frac{\left(F_*B \xrightarrow{q''} A \xrightarrow{q'} F_*B \right) = \text{id}_{F_*B}}{\left(F^*F_*B \xrightarrow{F^*q''} F^*A \xrightarrow{q} B \right) = \varepsilon_B}$$

As an extension of the projector splitting $\varphi = i \circ q$ along the isomorphism F^*q' and its inverse, the composite $L\varphi \circ \varepsilon$ is clearly a projector splitting. Hence (16).

Towards (17), consider the following split equalizer in $\mathbb{A}_{\vec{F}}^{\cup}$

$$\begin{array}{ccccc} \varphi & \xrightarrow{F^*\eta \circ \varphi} & F^*F_*\varphi & \xrightarrow{F^*F_*(F^*\eta \circ \varphi)} & F^*F_*F^*F_*\varphi \\ \nwarrow_{\varphi \circ \varepsilon} & & \nwarrow_{F^*\eta \circ F^*F_*\varphi} & & \nwarrow_{F^*F_*\varphi \circ \varepsilon} \end{array} \quad (19)$$

Splitting the projectors in \mathbb{B} yields the following split equalizer

$$\begin{array}{ccccc}
B & \xrightarrow{L\varphi} & F^*F_*B & \xrightarrow{F^*F_*L\varphi} & F^*F_*F^*F_*B \\
\swarrow \varepsilon & & & \searrow F^*\eta & \\
& & & & \\
& & & \nwarrow \varepsilon &
\end{array} \quad (20)$$

which gives (17).

The same reasoning applied to ψ and its splitting in (15) shows that $L\psi$, as defined there, is also an \overrightarrow{F} -coalgebra. Combining (18) with the analogous diagram for $F_*\psi$, splitting into γ and $\overline{\gamma}$, furthermore gives

$$F_*Lf = p' \circ f \circ q''$$

The definition of Lf in (15) then displays the equation $\vec{F}f \circ L\varphi = L\psi \circ f$, which means that Lf is an \vec{F} -coalgebra homomorphism from $L\varphi$ to $L\psi$. This completes the proof that the functors R and L are well defined.

To see that the counit $e : LR \rightarrow \text{Id}$ is a natural isomorphism, set in (15) $\varphi = R\beta$ and $A = F_*B$, which makes $q' = q'' = \text{id}$. Since the definition of R gives the splitting $R\beta = \beta \circ \varepsilon$, and the definition of L says that $LR\beta$ is the monic part of that splitting (followed by F^*q' , which is now identity), we have $LR\beta = \beta$. The counit $e : LR \rightarrow \text{Id}$ is thus the identity.

The fact that $h : \text{Id} \rightarrow RL$ is a natural isomorphism can be seen on (15), which displays not only φ and $L\varphi = F^*q' \circ i$, but also $RL\varphi = L\varphi \circ \varepsilon = (F^*q' \circ i) \circ (q \circ F^*q'') = F^*q' \circ \varphi \circ F^*q''$. The isomorphism $h_\varphi \in \mathbb{A}_F^{\leftarrow}(\varphi, RL\varphi)$ is thus $h_\varphi = \left(A \xrightarrow{q'} F_*B\right)$ in \mathbb{A} . The fact that it is consistent, i.e. an $\mathbb{A}_F^{\leftarrow}$ -morphism, is clear from the following commutative diagram.

$$\begin{array}{ccc}
F^*A & \xrightarrow{h_\varphi = F^*q'} & F^*F_*B \\
\varphi \downarrow & & \downarrow F^*q'' \\
& & F^*A \\
& & \downarrow \varphi \\
& & F^*A \\
& & \downarrow F^*q' \\
F^*A & \xrightarrow{F^*q'} & F^*F_*B
\end{array}
\quad \begin{array}{l} \\ \\ \\ RL\varphi \\ \\ \end{array}
\quad (21)$$

This completes the proof that $L \dashv R : \mathbb{B}^{\vec{F}} \rightarrow \mathbb{A}_F^{\leftrightarrow}$ is an equivalence of categories. \square

3.3 Injective coalgebras as algebras

As it is usually the case with algebras and coalgebras, the dual constructions are symmetric, but their interpretations and concrete applications are quite different. For the moment, we just spell out the dual structures and propositions, and leave the dual proofs as an exercise.

While every algebra is a quotient of a free algebra, and projective algebras are also subalgebras of free algebras, every coalgebra is a subalgebra of a cofree coalgebra, and *injective* coalgebras are also quotients of cofree coalgebras. The category of injective \vec{F} -coalgebras and compliant homomorphisms is thus

$$\begin{aligned} |\mathbb{B}_{\vec{F}}^\cup| &= \coprod_{X \in |\mathbb{B}|} \left\{ \varphi \in \mathbb{A}(F_*X, F_*X) \mid \varphi \circ \varphi = \varphi \wedge F^*F_*X \xrightarrow{F_*\varphi} X \right\} \\ \mathbb{B}_{\vec{F}}^\cup(\varphi, \psi) &= \left\{ h \in \mathbb{A}(F_*X, F_*Y) \mid \begin{array}{ccc} F_*X & \xrightarrow{h} & F_*Y \\ \downarrow \varphi & & \uparrow \psi \\ F_*X & \xrightarrow{h} & F_*Y \end{array} \right\} \end{aligned}$$

On the other hand, the category of injective coalgebras and consistent homomorphisms is

$$\begin{aligned} |\mathbb{B}_F^{\rightrightarrows}| &= \coprod_{X \in |\mathbb{B}|} \left\{ \varphi \in \mathbb{A}(F_*X, F_*X) \mid \varphi \circ \varphi = \varphi \right\} \\ \mathbb{B}_F^{\rightrightarrows}(\varphi, \psi) &= \left\{ f \in \mathbb{B}(X, Y) \mid \begin{array}{ccc} F_*X & \xrightarrow{F_*f} & F_*Y \\ \downarrow \varphi & & \downarrow \psi \\ F_*X & \xrightarrow{F_*f} & F_*Y \end{array} \right\} \end{aligned}$$

Theorem 3.3. *For every adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$, with the induced monad $\overleftarrow{F} = F_*F^*$ and comonad $\overrightarrow{F} = F^*F_*$, the category of \overleftarrow{F} -algebras is equivalent with the category of injective \overrightarrow{F} -algebras and consistent homomorphisms, provided that \mathbb{A} is Cauchy complete. The equivalence is given by the functors*

$$\mathbb{A}^{\overleftarrow{F}} \begin{array}{c} \xleftarrow{R} \\ \xrightarrow{L} \end{array} \mathbb{B}_F^{\rightrightarrows}$$

where the rules

$$\frac{\left(\overleftarrow{A} \xrightarrow{\alpha} A \right) \in |\mathbb{A}^{\overleftarrow{F}}|}{R\alpha = \left\langle F^*A, \overleftarrow{F}A \xrightarrow{\alpha} A \xrightarrow{\eta} \overleftarrow{F}A \right\rangle \in |\mathbb{B}_F^{\rightrightarrows}|}$$

and

$$\frac{\left(A \xrightarrow{f} C \right) \in \mathbb{A}^{\overleftarrow{F}}(\alpha, \gamma)}{Rf = \left(\overleftarrow{F}A \xrightarrow{\overleftarrow{F}f} \overleftarrow{F}C \right) \in \mathbb{B}_F^{\rightrightarrows}(R\alpha, R\gamma)}$$

define R , whereas the object part of L

$$\frac{\left\langle B, F_*B \xrightarrow{\varphi} F_*B \right\rangle \in |\mathbb{B}_F^{\rightrightarrows}|}{L\varphi = \left(F_*F^*A \xrightarrow{F_*\varphi'} F_*B \xrightarrow{q} A \right) \in |\mathbb{A}^{\overleftarrow{F}}|}$$

and its arrow part

$$\frac{(B \xrightarrow{g} D) \in \mathbb{B}_F^{\leftrightarrow}(\varphi, \psi)}{(A \xrightarrow{Lg} C) \in \mathbb{A}^{\overleftarrow{F}}(L\varphi, L\psi)}$$

are defined using the projector splittings in the following diagram

$$\begin{array}{ccccc}
& & L\varphi & & \\
& \swarrow & & \searrow & \\
F_*F^*A & \xrightarrow{\sim} & F_*B & \xrightarrow{q} & A & \xrightarrow{i} & F_*B & \xrightarrow{\sim} & F_*F^*A \\
\downarrow F_*F^*Lg & & \downarrow F_*g & & \downarrow Lg & & \downarrow F_*g & & \downarrow F_*F^*Lg \\
F_*F^*C & \xrightarrow{\sim} & F_*D & \xrightarrow{p} & C & \xrightarrow{j} & F_*D & \xrightarrow{\sim} & F_*F^*C \\
& \nwarrow & & \nearrow & & \nwarrow & & \nearrow & \\
& & L\psi & & \psi & & & &
\end{array} \tag{22}$$

4 Application

Interpreted along the lines of the example from Sec. 1.2, the category $\mathbb{C}_S^{\leftrightarrow}$ of projectors and consistent homomorphisms, induced by the state monad \overleftarrow{S} on a cartesian closed category \mathbb{C} , can be viewed as a model of data release policies. The idea is that

- a projector $(S \times A \xrightarrow{\varphi} S \times A) \in |\mathbb{C}_S^{\leftrightarrow}|$ filters private data $a \in A$ and private states $s \in S$ and releases a public pair

$$\varphi(s, a) = \langle \varphi_0(s, a), \varphi_1(s, a) \rangle \in S \times A$$

- a morphism $(A \xrightarrow{f} B) \in \mathbb{C}_S^{\leftrightarrow}(\varphi, \psi)$ can be thought of as a deterministic channel which maps data of type A to data of type B in such a way that the following requirements are satisfied

$$\psi_0(s, f(a)) = \varphi_0(s, a) \tag{23}$$

$$\psi_1(s, f(a)) = f(\varphi_1(s, a)) \tag{24}$$

where $(S \times B \xrightarrow{\psi} S \times B) \in |\mathbb{C}_S^{\leftrightarrow}|$ is another policy.

Conditions (23–24) guarantee that the channel f behaves consistently with the policies φ and ψ . Note that this is a special case of the model from Sec. 1.2, in the sense that we are not capturing the consistency of a database $S \times A \xrightarrow{g} S \times B$ with the policies $S \times A \xrightarrow{\varphi} S \times A$ and $S \times B \xrightarrow{\psi} S \times B$, but only the consistency of a stateless channel $A \xrightarrow{f} B$.

If we accept this restriction for a moment, the equivalence $\mathbb{C}_S^{\leftrightarrow} \simeq \mathbb{C}^{\overrightarrow{S}}$ provides an interesting characterization of data release policies, with the consistent channels as the

morphisms between them: they are equivalent to coalgebras for the comonad

$$\begin{aligned} \overrightarrow{S} : \mathbb{C} &\rightarrow \mathbb{C} \\ X &\mapsto S \times (S \Rightarrow X) \end{aligned} \quad (25)$$

with the coalgebra homomorphisms. More precisely, a projective \overleftarrow{S} -algebra $S \times A \xrightarrow{\varphi} S \times A$, which we viewed as a Mealy machine [26, Sec. 2.7(a)]

$$S \times A \xrightarrow{\varphi_0} S \quad S \times A \xrightarrow{\varphi_1} A \quad (26)$$

induces a coalgebra $B \xrightarrow{\beta} S \times (S \Rightarrow B)$, which boils down to of a pair of maps reminding of a Moore machine [26, Sec. 2.7(b)]

$$B \xrightarrow{\beta_0} S \quad B \times S \xrightarrow{\beta_1} B \quad (27)$$

It is conspicuous, however, that the state space S of the Mealy machine φ has become the alphabet in the corresponding Moore machine $\beta = L\varphi$, where L is the functor from Thm. 3.2. The state space B of the Moore machine β arises from the construction of $L\varphi$ in (15) as the set of public pairs:

$$B = \{ \langle s, a \rangle \in S \times A \mid \varphi(s, a) = \langle s, a \rangle \}$$

Note, however, that both machines are of a very special kind: the Mealy machine is idempotent, and the Moore machine satisfies the coalgebra conditions

$$\varepsilon \circ \beta = \text{id} \quad \overrightarrow{S}\beta \circ \beta = \nu \circ \beta$$

which for the components in (27) correspond to the following equations

$$\begin{aligned} \beta_0(\beta_1(b, s)) &= s \\ \beta_1(b, \beta_0(b)) &= b \\ \beta_1(\beta_1(b, s), t) &= \beta_1(b, t) \end{aligned}$$

In a sense (formalized by Thm. 3.2), these equations realize on the set of public pairs B precisely the data filtering condition that was realized on the set of all pairs $S \times A$ by the idempotence of φ .

To go beyond the stateless morphisms, and capture not just channels in the form $A \xrightarrow{f} B$, but also databases in the form $S \times A \xrightarrow{g} S \times B$, consider the adjunction

$$\mathbb{C} \begin{array}{c} \xrightarrow{S^b} \\ \perp \\ \xleftarrow{S_b} \end{array} \mathbb{C}_{\overleftarrow{S}} \quad (28)$$

where

$$\begin{aligned} S^b X &= X & S_b X &= \overleftarrow{S} X \\ S^b f &= (S \times f) & S_b f &= (S \Rightarrow f) \end{aligned}$$

The category of injective coalgebras in $\mathbb{C}_{\overleftarrow{S}}$ is now as follows:

$$|\mathbb{C}_{\overleftarrow{S}}^{\leftrightarrow}| = \coprod_{A \in |\mathbb{C}|} \left\{ \varphi \in \mathbb{C}(\overleftarrow{S}A, \overleftarrow{S}A) \mid \varphi \circ \varphi = \varphi \right\}$$

$$\mathbb{C}_{\overleftarrow{S}}^{\leftrightarrow}(\varphi, \psi) = \left\{ g \in \mathbb{C}(S \times A, S \times B) \mid \begin{array}{ccc} \overleftarrow{S}A & \xrightarrow{S \Rightarrow g} & \overleftarrow{S}B \\ \downarrow \varphi & & \downarrow \psi \\ \overleftarrow{S}A & \xrightarrow{S \Rightarrow g} & \overleftarrow{S}B \end{array} \right\}$$

The consistent homomorphisms $S \times A \xrightarrow{g} S \times B$ can now be construed as databases. The policies with which they are consistent are more general than those considered so far. Policies $S \times A \rightarrow S \times A$ in $\mathbb{C}_S^{\leftrightarrow}$ filtered private states and data and supplied the corresponding public pairs. Policies $\overleftarrow{S}A \rightarrow \overleftarrow{S}A$ in $\mathbb{C}_{\overleftarrow{S}}^{\leftrightarrow}$ filter entire stateful behaviors.

Interestingly, though, Thm. 3.3 provides the equivalence $\mathbb{C}_{\overleftarrow{S}}^{\leftrightarrow} \simeq \mathbb{C}^{\overleftarrow{S}}$, and one of its corollaries⁵ provides an equivalence $\mathbb{C}^{\overleftarrow{S}} \simeq \mathbb{C}$. The equivalences are nontrivial, and may require further research. They suggest that implementing policies within a model of data release, and using these policies to filter out the private data, and to extract the public data alone, leads to an equivalent model, but this time consisting of the public data alone. Filtering out the private data can thus be formalized as an equivalence. Privacy policies can be formalized to make the publicly released data structurally indistinguishable from all data.

5 Related and further work

Although the presented constructions emerged within a practice-driven effort towards modeling and analyzing data release policies using the salient tools of monadic programming, the research path led through the realm of basic monad theory, with some old questions still lurking, and with the theoretic repercussions surpassing not only our practical goals, but probably also our current understanding. Back in 1968, in the first of the Batelle volumes, Barr [9] raised the question of comonadicity of the left adjoint of a monadic functor. More precisely, he considered the adjunctions in the form

$$\mathbb{A} \begin{array}{c} \xrightarrow{T^\sharp} \\ \perp \\ \xleftarrow{T_\sharp} \end{array} \mathbb{A}^{\overleftarrow{T}} \quad (29)$$

and asked under which conditions would the functor T^\sharp be comonadic. This means that the comparison functor

$$\mathbb{A} \longrightarrow \left(\mathbb{A}^{\overleftarrow{T}} \right)^{\overrightarrow{T}}$$

should be an equivalence, where $\overrightarrow{T} = T^\sharp T_\sharp$. Barr provided the answer for the special cases when \mathbb{A} is the category of sets, pointed or not, and when it is the category of

⁵to be presented elsewhere

vector spaces and linear operators. He suggested that the general answer might be difficult. The question seems to have been reemerging regularly in various guises, most recently in Jacobs' work on coalgebras over algebras as an abstract form of the concept of basis [27], extending the results of [18] about bases as Sweedler-style coalgebras to bases as coalgebras for comonads.

Theorems 3.2 provides the equivalence $\mathbb{B}^{\vec{F}} \simeq \mathbb{A}_F^{\leftarrow}$ for *any* resolution $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$ of the monad \overleftarrow{F} . More precisely, if besides the adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$ there is also $G^* \dashv G_* : \mathbb{C} \rightarrow \mathbb{A}$, and the monads $G_*G^* \cong F_*F^*$ are isomorphic to a monad $T : \mathbb{A} \rightarrow \mathbb{A}$, coherently with respect to the monad structures [10, Sec. 3.6], then

$$\mathbb{B}^{\vec{F}} \simeq \mathbb{A}_T^{\leftarrow} \simeq \mathbb{C}^{\vec{G}}$$

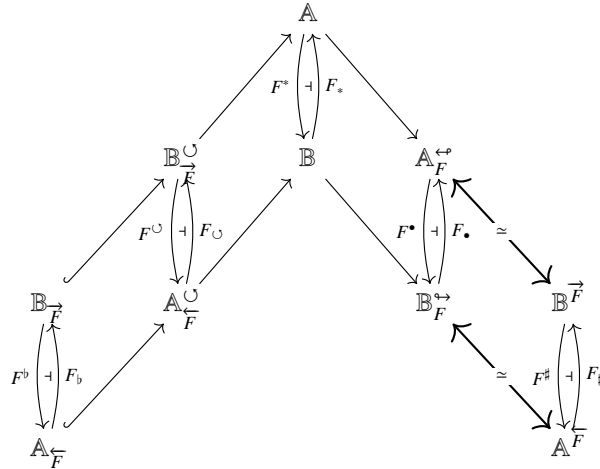
Instantiating to the adjunction displayed in (29) gives

$$\left(\mathbb{A}^{\overleftarrow{T}}\right)^{\vec{T}} \simeq \mathbb{A}_T^{\leftarrow}$$

The question of comonadicity of T^\sharp can thus be studied on the comparison functor

$$\mathbb{A} \longrightarrow \mathbb{A}_T^{\leftarrow}$$

which the reader may enjoy as an exercise. In a similar way, Beck's General Monadicity Theorem [12, 10, Thm. 3.3.13] can be stated and proved by unravelling the projectors from behind the split coequalizers in the original formulation. In general, the projector view of algebras and coalgebras, opened by Theorems 3.2 and 3.3, seems to facilitate analyses of monadicity, and even enable analysis of relative monadicity [4, 5]. On the other hand, it opens up an alley towards classifying resolutions in general, as illustrated in the following diagram.



Of special interest here is the adjunction $(F^\bullet \dashv F_\bullet) : \mathbb{B}_F^{\rightrightarrows} \rightarrow \mathbb{A}_F^{\leftarrow\rightarrow}$ defined by

$$\frac{\langle A, F^* A \xrightarrow{\varphi} F^* A \rangle \in |\mathbb{A}_F^{\leftarrow\rightarrow}|}{\langle F^* F_* B_\varphi, F_* F^* F_* B_\varphi \xrightarrow{F_* \varepsilon} F_* B_\varphi \xrightarrow{\eta} F_* F^* F_* B_\varphi \rangle \in |\mathbb{B}_F^{\rightrightarrows}|}$$

and

$$\frac{\langle D, F_* D \xrightarrow{\psi} F_* D \rangle \in |\mathbb{B}_F^{\rightrightarrows}|}{\langle F_* F^* C_\psi, F^* F_* F^* C_\psi \xrightarrow{\varepsilon} F^* C_\psi \xrightarrow{F^* \eta} F^* F_* F^* C_\psi \rangle \in |\mathbb{A}_F^{\leftarrow\rightarrow}|}$$

where $F^* A \twoheadrightarrow B_\varphi \rightarrowtail F^* A$ and $F_* D \twoheadrightarrow C_\psi \rightarrowtail F_* D$ are the splittings of φ and of ψ , respectively. The adjunction $(F^\bullet \dashv F_\bullet) : \mathbb{B}_F^{\rightrightarrows} \rightarrow \mathbb{A}_F^{\leftarrow\rightarrow}$ is the *nucleus* of the adjunction $F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A}$ [37, 38, 30, 29, 43, 44]. The fact that the functor F_\bullet is monadic, and the functor F^\bullet is comonadic will be proved in the full version of this paper.

References

- [1] Michael Gordon Abbott, Thorsten Altenkirch, Neil Ghani, and Conor McBride. Constructing polymorphic programs with quotient types. In Dexter Kozen and Carron Shankland, editors, *Mathematics of Program Construction, 7th International Conference, MPC 2004, Stirling, Scotland, UK, July 12-14, 2004, Proceedings*, volume 3125 of *Lecture Notes in Computer Science*, pages 2–15. Springer, 2004.
- [2] Danel Ahman, James Chapman, and Tarmo Uustalu. When is a container a comonad? *Logical Methods in Computer Science*, 10(3), 2014.
- [3] Thorsten Altenkirch. Extensional equality in intensional type theory. In *14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2-5, 1999*, pages 412–420. IEEE Computer Society, 1999.
- [4] Thorsten Altenkirch, James Chapman, and Tarmo Uustalu. Relative monads formalised. *J. Formalized Reasoning*, 7(1):1–43, 2014.
- [5] Thorsten Altenkirch, James Chapman, and Tarmo Uustalu. Monads need not be endofunctors. *Logical Methods in Computer Science*, 11(1), 2015.
- [6] Thorsten Altenkirch and Ambrus Kaposi. Type theory in type theory using quotient inductive types. In Rastislav Bodík and Rupak Majumdar, editors, *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016*, pages 18–29. ACM, 2016.
- [7] Michael Artin, Alexander Grothendieck, and Jean-Louis Verdier, editors. *Séminaire de Géométrie Algébrique: Théorie des Topos et Cohomologie Étale des Schemas (SGA 4)*, number 269,270,305 in *Lecture Notes in Mathematics*. Springer-Verlag, 1964. Second edition, 1972.

- [8] Steven Awodey, Nicola Gambino, and Kristina Sojakova. Inductive types in homotopy type theory. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, pages 95–104. IEEE Computer Society, 2012.
- [9] Michael Barr. Coalgebras in a category of algebras. In *Category Theory, Homology Theory and their Applications I*, volume 86 of *Lecture Notes in Mathematics*, pages 1–12. Springer Berlin Heidelberg, 1969.
- [10] Michael Barr and Charles Wells. *Toposes, Triples, and Theories*. Number 278 in Grundlehren der mathematischen Wissenschaften. Springer-Verlag, 1985. Republished in: Reprints in Theory and Applications of Categories, No. 12 (2005) pp. 1-287.
- [11] Gilles Barthe, Venanzio Capretta, and Olivier Pons. Setoids in type theory. *J. Funct. Program.*, 13(2):261–293, 2003.
- [12] Jonathan Mock Beck. *Triples, Algebras and Cohomology*. PhD thesis, Columbia University, 1967. Reprinted in *Theory and Applications of Categories*, No. 2, 2003, pp. 1–59.
- [13] Nick Benton. Categorical monads and computer programming. *LMS Impact150 Stories*, 1:9–13, 2015.
- [14] Nick Benton, John Hughes, and Eugenio Moggi. Monads and effects. In Gilles Barthe, Peter Dybjer, Luis Pinto, and João Saraiva, editors, *Applied Semantics, International Summer School, APPSEM 2000, Caminha, Portugal, September 9-15, 2000, Advanced Lectures*, volume 2395 of *Lecture Notes in Computer Science*, pages 42–122. Springer, 2000.
- [15] Marcello M. Bonsangue, Jan J. M. M. Rutten, and Alexandra Silva. Coalgebraic logic and synthesis of mealy machines. In Roberto M. Amadio, editor, *Foundations of Software Science and Computational Structures, 11th International Conference, FOSSACS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29 - April 6, 2008. Proceedings*, volume 4962 of *Lecture Notes in Computer Science*, pages 231–245. Springer, 2008.
- [16] Francis Borceux. *Handbook of Categorical Algebra*. Number 50 in Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1994. Three volumes.
- [17] S. Brookes and S. Geva. Computational comonads and intensional semantics. In M.P. Fourman, P.T. Johnstone, and A.M. Pitts, editors, *Categories in Computer Science*, pages 1–44. Cambridge University Press, 1992.
- [18] Bob Coecke, Dusko Pavlovic, and Jamie Vicary. A new description of orthogonal bases. *Math. Structures in Comp. Sci.*, 23(3):555–567, 2013. arxiv.org:0810.0812.

- [19] Roger Godement. *Topologie Algébrique et Théorie des Faisceaux*. Hermann, 1958.
- [20] Joseph A Goguen, James W Thatcher, Eric G Wagner, and Jesse B Wright. An introduction to categories, algebraic theories and algebras. Technical report, IBM Thomas J. Watson Research Division, 1975.
- [21] Joseph A Goguen, James W Thatcher, Eric G Wagner, and Jesse B Wright. Initial algebra semantics and continuous algebras. *Journal of the ACM (JACM)*, 24(1):68–95, 1977.
- [22] G. Grätzer. *Universal Algebra*. Mathematics and Statistics. Springer New York, 2008.
- [23] Helle Hvid Hansen, David Costa, and Jan J. M. M. Rutten. Synthesis of mealy machines using derivatives. *Electr. Notes Theor. Comput. Sci.*, 164(1):27–45, 2006.
- [24] Martin Hofmann. A simple model for quotient types. In Mariangiola Dezani-Ciancaglini and Gordon D. Plotkin, editors, *Typed Lambda Calculi and Applications, Second International Conference on Typed Lambda Calculi and Applications, TLCA '95, Edinburgh, UK, April 10-12, 1995, Proceedings*, volume 902 of *Lecture Notes in Computer Science*, pages 216–234. Springer, 1995.
- [25] Martin Hofmann and Thomas Streicher. The groupoid model refutes uniqueness of identity proofs. In *Proceedings of the Ninth Annual Symposium on Logic in Computer Science (LICS '94), Paris, France, July 4-7, 1994*, pages 208–212. IEEE Computer Society, 1994.
- [26] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation*. Addison-Wesley, 1979.
- [27] Bart Jacobs. Bases as coalgebras. *Logical Methods in Computer Science*, 9(3), 2013.
- [28] Peter Johnstone. *Stone Spaces*. Number 3 in Cambridge Studies in Advanced Mathematics. Cambridge University Press, 1982.
- [29] Toshiki Kataoka and Dusko Pavlovic. Nuclei of adjunctions, December 2015. In progress.
- [30] Toshiki Kataoka and Dusko Pavlovic. Towards Concept Analysis in Categories: Limit Inferior as Algebra, Limit Superior as Coalgebra. In Lawrence S. Moss and Pawel Sobocinski, editors, *Proceedings of CALCO 2015*, volume 35 of *LIPICs*, pages 130–155, Dagstuhl, Germany, 2015. Leibniz-Zentrum für Informatik. [arxiv.org:1505.01098](https://arxiv.org/abs/1505.01098).
- [31] Yoshiaki Kinoshita and John Power. Category theoretic structure of setoids. *Theor. Comput. Sci.*, 546:145–163, 2014.

- [32] Heinrich Kleisli. Every standard construction is induced by a pair of adjoint functors. *Proceedings of the American Mathematical Society*, pages 544–546, 1965.
- [33] Fred EJ Linton. Some aspects of equational categories. In *Proceedings of the Conference on Categorical Algebra held in La Jolla*, pages 84–94. Springer, 1966.
- [34] Ernest Manes. *Algebraic Theories*. Number 26 in Graduate Texts in Mathematics. Springer-Verlag, 1976.
- [35] Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93:55–92, 1991.
- [36] Robert Paré. On absolute colimits. *J. Alg.*, 19:80–95, 1971.
- [37] Dusko Pavlovic. Quantitative Concept Analysis. In Florent Domenach, Dmitry I. Ignatov, and Jonas Poelmans, editors, *Proceedings of ICFCA 2012*, volume 7278 of *Lecture Notes in Artificial Intelligence*, pages 260–277. Springer Verlag, 2012. arXiv:1204.5802.
- [38] Dusko Pavlovic. Bicompletions of distance matrices. In Bob Coecke, Luke Ong, and Prakash Panangaden, editors, *Computation, Logic, Games and Quantum Foundations. The Many Facets of Samson Abramsky*, volume 7860 of *Lecture Notes in Computer Science*, pages 291–310. Springer Verlag, 2013.
- [39] Dusko Pavlovic and Peter-Michael Seidel. Privacy networks: Trust, noninterference, differential privacy. Technical report, ASECOLab, February 2016. working paper, available at dusko.org.
- [40] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. homotopytypetheory.org/book, Institute for Advanced Study, 2013.
- [41] Tarmo Uustalu and Varmo Vene. Comonadic notions of computation. *Electr. Notes Theor. Comput. Sci.*, 203(5):263–284, 2008.
- [42] Philip Wadler. Monads for functional programming. In Johan Jeuring and Erik Meijer, editors, *Advanced Functional Programming, First International Spring School on Advanced Functional Programming Techniques, Båstad, Sweden, May 24-30, 1995, Tutorial Text*, volume 925 of *Lecture Notes in Computer Science*, pages 24–52. Springer, 1995.
- [43] Simon Willerton. Tight spans, isbell completions and semi-tropical modules. *Theory and Applications of Categories*, 28(22):696–732, 2013.
- [44] Simon Willerton. The Legendre-Fenchel transform from a category theoretic perspective, 2015. arXiv:1501.03791.

A Appendix: Adjunctions, monads, comonads

Definition A.1. An adjunction $F = (F^* \dashv F_* : \mathbb{B} \rightarrow \mathbb{A})$ consists of the following data

- categories \mathbb{A} and \mathbb{B}
- functors $F^* : \mathbb{A} \rightarrow \mathbb{B}$ and $F_* : \mathbb{B} \rightarrow \mathbb{A}$, called *left adjoint* and *right adjoint*, respectively
- natural transformations $\eta : \text{id}_{\mathbb{A}} \rightarrow F_* F^*$ and $\varepsilon : F^* F_* \rightarrow \text{id}_{\mathbb{B}}$, called the *adjunction unit* and *counit*, respectively,

such that

Definition A.2. A monad (T, η, μ) consists of the following data

- category \mathbb{A}
- functor $T : \mathbb{A} \rightarrow \mathbb{A}$
- natural transformations $\eta : \text{id}_{\mathbb{A}} \rightarrow T$ and $\mu : TT \rightarrow T$, called, respectively, the *monad unit* and *evaluation* (or *cochain*)

which satisfy the following conditions

A commutative diagram for monad multiplication. It shows three stages of a diagram separated by equals signs. The first stage has a square with nodes \mathbb{A} (top-left), \mathbb{A} (top-right), $T\mathbb{A}$ (bottom-left), and $T\mathbb{A}$ (bottom-right). Arrows are: η (curved, top-left to top-right), T (diagonal, top-left to bottom-right), μ (horizontal, top-left to top-right), and T (vertical, top-right to bottom-right). The second stage is a single vertical arrow T from \mathbb{A} to $T\mathbb{A}$. The third stage is a square with nodes \mathbb{A} (top-left), \mathbb{A} (top-right), $T\mathbb{A}$ (bottom-left), and $T\mathbb{A}$ (bottom-right). Arrows are: T (diagonal, top-left to bottom-right), μ (horizontal, top-left to top-right), T (vertical, top-right to bottom-right), and η (curved, bottom-left to bottom-right).

Definition A.3. A comonad (S, ε, ν) consists of the following data

- category \mathbb{A}
- functor $S : \mathbb{A} \rightarrow \mathbb{A}$
- natural transformations $\varepsilon : S \rightarrow \text{id}_{\mathbb{A}}$ and $\nu : S \rightarrow SS$, called the *comonad counit* and *coevaluation* (or *chain*), respectively,

which satisfy the following conditions

A commutative diagram for comonad coevaluation. It shows two stages separated by an equals sign. The first stage has a square with nodes \mathbb{A} (top-left), \mathbb{A} (top-right), \mathbb{A} (bottom-left), and $S\mathbb{A}$ (bottom-right). Arrows are: S (horizontal, top-left to top-right), ν (diagonal, top-left to bottom-right), S (vertical, top-right to bottom-right), and S (diagonal, bottom-left to bottom-right). The second stage has a square with nodes \mathbb{A} (top-left), \mathbb{A} (top-right), \mathbb{A} (bottom-left), and \mathbb{A} (bottom-right). Arrows are: S (horizontal, top-left to top-right), ν (diagonal, top-left to bottom-right), S (vertical, top-right to bottom-right), and S (horizontal, bottom-left to bottom-right).

A commutative diagram for comonad counit. It shows three stages of a diagram separated by equals signs. The first stage has a square with nodes \mathbb{A} (top-left), \mathbb{A} (top-right), \mathbb{A} (bottom-left), and $S\mathbb{A}$ (bottom-right). Arrows are: ε (curved, top-left to top-right), S (diagonal, top-left to bottom-right), ν (horizontal, top-left to top-right), and S (vertical, top-right to bottom-right). The second stage is a single vertical arrow S from \mathbb{A} to $S\mathbb{A}$. The third stage has a square with nodes \mathbb{A} (top-left), \mathbb{A} (top-right), \mathbb{A} (bottom-left), and \mathbb{A} (bottom-right). Arrows are: S (diagonal, top-left to bottom-right), ν (horizontal, top-left to top-right), S (vertical, top-right to bottom-right), and ε (curved, bottom-left to bottom-right).

Definition A.4. The Kleisli construction maps the monad $T : \mathbb{A} \rightarrow \mathbb{A}$ to the adjunction $\overleftarrow{\mathcal{K}}T = (T^b \dashv T_b : \mathbb{A}_{\overleftarrow{\mathcal{K}}T} \rightarrow \mathbb{A})$ where the category $\mathbb{A}_{\overleftarrow{\mathcal{K}}T}$ consists of

- *free algebras* as objects, which boil down to $|\mathbb{A}_{\overleftarrow{\mathcal{K}}T}| = |\mathbb{A}|$;

- *algebra homomorphisms* as arrows, which boil down to $\mathbb{A}_{\tilde{T}}(x, x') = \mathbb{A}(x, Tx')$;

with the composition

$$\begin{aligned} \mathbb{A}_{\tilde{T}}(x, x') \times \mathbb{A}_{\tilde{T}}(x', x'') &\xrightarrow{\circ} \mathbb{A}_{\tilde{T}}(x, x'') \\ \langle x \xrightarrow{f} Tx', x' \xrightarrow{g} Tx'' \rangle &\mapsto (x \xrightarrow{f} Tx' \xrightarrow{Tg} TTx'' \xrightarrow{\mu} Tx'') \end{aligned}$$

and with the identity on x induced by the monad unit $\eta : x \rightarrow Tx$

Definition A.5. The Eilenberg-Moore construction maps the monad $T : \mathbb{A} \rightarrow \mathbb{A}$ to the adjunction $\tilde{\mathcal{E}}T = (T^\# \dashv T_\# : \mathbb{A}^{\tilde{T}} \rightarrow \mathbb{A})$ where the category $\mathbb{A}^{\tilde{T}}$ consists of

- *all algebras* as objects:

$$|\mathbb{A}^{\tilde{T}}| = \sum_{x \in |\mathbb{A}|} \{ \alpha \in \mathbb{A}(Tx, x) \mid \alpha \circ \eta = \text{id} \wedge \alpha \circ T\alpha = \alpha \circ \mu \}$$

- *algebra homomorphisms* as arrows:

$$\mathbb{A}^{\tilde{T}}(Tx \xrightarrow{\alpha} x, Tx' \xrightarrow{\gamma} x') = \{ f \in \mathbb{A}(x, x') \mid f \circ \alpha = \gamma \circ Tf \}$$

B Appendix: Split equalizers

Proposition B.1. Consider the diagram

$$\begin{array}{ccccc} A & \xrightarrow{i} & B & \xrightarrow{f} & C \\ & \searrow q & & \nwarrow j & \\ & & & & r \end{array}$$

where

$$q \circ i = \text{id}_A \quad r \circ j = \text{id}_B \quad f \circ r \circ f = j \circ r \circ f$$

Then

- $r \circ f$ is idempotent and
- i is the equalizer of f and j if and only if $i \circ q = r \circ f$.